Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844
Vol. VI (2011), No. 3 (September), pp. 473-502

Human-inspired Identification of High-level Concepts using OWA and Linguistic Quantifiers

M.Z. Reformat, R.R. Yager, Z. Li, N. Alajlan

Marek Z. Reformat
thinkS^2: thinking Software and Systems laboratory
Electrical and Computer Engineering
University of Alberta, Canada
E-mail: Marek.Reformat@ualberta.ca

Ronald R. Yager
1. Machine Intelligence Institute
Iona Collage, New Rochelle, NY, USA
E-mail: yager@panix.com
2. Visiting Distinguished Scientist
King Saud University, Riyadh, Saudi Arabia

Zhan Li
thinkS^2: thinking Software and Systems laboratory
Electrical and Computer Engineering
University of Alberta, Canada
E-mail: zhanli@ualberta.ca

Naif Alajlan
Advanced Lab for Intelligent Systems Research
College of Computer and Information Sciences
King Saud University, Riyadh, Saudi Arabia
E-mail: najlan@ksu.edu.sa

Abstract: Intelligent agent based system can be used to identify high-level concepts matching sets of keywords provided by users. A new human-inspired approach to concept identification in documents is introduced here. The proposed method takes keywords and builds concept structures based on them. These concept structures are represented as hierarchies of concepts (HofC). The ontology is used to enrich HofCs with terms and other concepts (sub-concepts) based on concept definitions, as well as with related concepts. Additionally, the approach uses levels of importance of terms defining the concepts. The levels of importance of terms are continuously updated based on a flow of documents using an Adaptive Assignment of Term Importance (AATI) schema. The levels of activation of concepts identified in a document that match these in the HofC are estimated using ordered weighted averaging (OWA) operators with linguistic quantifiers. A simple case study presented in the paper is designed to illustrate the approach.
Keywords: concept identification, text documents, ontology, hierarchy of concepts, ordered weighted averaging operator, importance of concepts.

1 Introduction

Keyword-matching is the most popular approach used in current search engines for finding useful texts [1]. However, the meaning of words used as keywords is context-dependent. In

addition, users tend to use very few keywords (three or less) what created difficulties in identifying their proper meaning [2, 3].

In this paper we focus on human-like analysis of text, i.e., processes of identification of concepts based on terms/words from a document. The proposed approach reflects a real life scenario where a person links the information in a document to a specific topic by recognizing multiple terms that describe concepts associated with this topic. The terms that appear in a document "activate" terms known to the person that are related to the topic's concepts. In such a way, a net of relevant words is activated. The more words related to a specific concept are activated, the more confidence the person has that the document contains this specific concept. In other words, in concept-based models, we try to "understand" the meaning of a set of keywords and attempt to determine presence of concepts defined by these keywords in documents.

The proposed approach starts with a simple set of keywords provided by the user and representing an entity/item the user is interested in. This set is transformed into a Hierarchy of Concepts (HofC). The HofC is further enhanced with pieces of relevant knowledge obtained from ontology-based knowledge base. This ontology-based knowledge base is created based on ontology domains. Its elements are equipped with values representing levels of contribution of individual keywords towards definitions of concepts. These values are automatically determined using AATI – a schema proposed in ... for determining importance values of keywords based on continuous flow of documents.

The proposed approach relies on the following idea: a process of identification of a concept in a document is equivalent to determining a level of activation of HofC representing this concept. Once a HofC is built and enhanced using an ontology-based knowledge base, it is "checked against" a document. The words found in document are matched with terms and concepts from HofC. Every time a term/concept is found it is activated. The activation levels of terms and concepts are aggregated using Ordered Weighted Averaging (OWA) operator. The weights of OWA are determined based on activation levels and linguistic quantifiers, such as *MOST*, *AT LEAST HALF*, *ALL* and *ABOUT ONE THIRD*. The activation and aggregation propagate in a bottom-up fashion. A final activation level – level of activation of the HofC – is treated as an indicator of how well the concept is present in the document.

The paper is organized in the following way. Section 2 contains a brief description of work done so far in the area of concept-based analysis of text with emphasis on applications of ontology to support definition and identification of concepts. The background is presented in Section 3. Section 4 contains an overview of the proposed approach. The next three section are dedicated to specific elements of that approach: the construction of ontology-based knowledge base is discussed in Section 5, Section 6 contains description of the process of building and enhancing HofCs, while the process of determining activation level of HofC for a document is described in Section 7. A case study with two different linguistic quantifiers is included in Section 8. The papers finishes with conclusions.

2 Related work

There are two most popular approached for concept-based analysis of documents: classifier-based approach, and concept structure-based approach. Because our work is focused on a concept-identification method that involved ontology we do not describe the classifier-based techniques for identification of concepts. Some example of the work related to this topic can be found in [4], [5], [6], and [7].

2.1 Concept Knowledge Bases

It can be noted that classifier-based techniques requires human-labor to set categories, build training sets, update models, etc. Therefore, constructing a background concept structure as a knowledge base to define concepts is a more suitable approach. Some example of such structures are:

- **Synonym thesauri** are treated as repositories of terms"related" to keywords provided by users. For example, Anick [8] has proposed a system that automatically generates an extended condition: "a boolean expression is composed by ORing each query term with any stored synonyms and then ANDing these clusters together." That is, the whole query is ORed together. Each OR term is composed of synonyms from an online thesaurus.

- **Conceptual taxonomy** is seen as a hierarchical organization of concepts. Each concept in a conceptual taxonomy connects to both its superconcepts and its subconcepts. Therefore, it provides a topological structure for efficient conceptual search and retrieval. In the project by Sun Microsystems, a conceptual indexing technique was proposed to automatically generate conceptual taxonomies [9].

- **Ontology** is a model of a domain or a problem. Ontologies can be used to provide formal semantics (meanings, concept-based information) to any sort of information, such as databases, Web documents, etc. A commonly accepted definition states that "an ontology is an explicit and formal specification of a conceptualization of a domain of interest" [10]. In general, an ontology is a representation of a set of concepts and relations between those concepts in a domain.

Ontologies are gaining popularity because of their applications to text analysis. Some of the popular ontologies are: WordNet, SENSUS, and Gene Ontology.

WordNet is an English lexical ontology built at Princeton University, which includes explanations of the terms and relations between terms (synonyms, antonyms, etc.) [11]. As the most popular linguistic ontology, WordNet is used by many researchers for expanded queries [12] [13]. However, the systems applying WordNet suffered from word sense disambiguation (WSD) because of polysemy[1]. This problem has been addressed in [14], [15], and [16]. Additionally, researchers report difficulties in applying linguistic-based ontologies to non-linguistic applications [18].

SENSUS is a natural language-based ontology developed by the Natural Language Group at Information Sciences Institute (ISI) [17]. It is an extension and reorganization of WordNet. It has been used for concept-based retrieval from online yellow pages and product catalogs [18].

Gene Ontology is composed of three structured controlled vocabularies describing gene products with their associated biological processes, cellular components and molecular functions [19]. This ontology is a part of an information retrieval system, KiPar, to facilitate access to the literature relevant to kinetic modelling of a given metabolic pathway in yeast [20].

2.2 Ontology-based Concept Identification

There are different ways to identify concepts in a text using ontology as a knowledge structure. We have divided those approaches into four major categories.

[1]Polysemy means more than one words with the same meanings

Regular Expressions, Rules, and Ontology

In this type of approaches, regular expressions are used to identify concepts from texts. Several predefined rules are then used to aggregate the identifications of single concepts.

Embley has proposed the use of information extraction ontologies, which are formalized over regular expressions [21]. Muller et al. have constructed an ontology-based information retrieval and extraction system for biological literature, which is called *Textpresso* [22] where biological concepts (e.g., gene, allele, cell or cell group, phenotype, etc.) are presented as regular expressions.

Natural Language Processing (NLP) with Ontology

Informally, NLP aims to solve problems by making computers understand and process human language. There are two types of approach in NLP: *deep* approach and *shallow* approach. As Styltsvig stated, "Deep approaches presume access to a comprehensive body of world knowledge. These approaches are not very successful in practice, mainly because access to such a body of knowledge does not exist, except in very limited domains" [23]. Shallow NLP techniques normally rely on simple rules to perform analysis.

Cimiano et al. have presented the LexOnto model consisting of a domain ontology and a corresponding ontology for associating lexical information to entities of the given domain ontology [24]. Morneau et al. have proposed SeseiOnto using NLP to identify concepts from texts [25]. Compared with the LexOnto model, this approach involves more shallow NLP. First of all, SeseiOnto removes articles and prepositions from NLP-based user queries (sentences) by tagging. The remaining words in the queries are matched to the concepts in the ontology.

Utilizing NLP and an ontology may be the most intuitive application to build concept-based systems. However, the discussions on how much NLP should be involved never stops. In LexiOnto [24], NLP exists in the entire process; while in SeseiOnto [25], NLP handles conversion from texts to concept-based structures. Though the NLP techniques allow computer agents to recognize concepts from texts, their complexity cannot be ignored.

Vector Space and Ontology

This type of solution is the most popular. In general, the ontology is utilized to expand the query, and vector space approach is used to evaluate similarity.

Vallet et al. have proposed an ontology based information retrieval system applying a vector-space model to retrieve relevant documents [26]. After terms and concepts are connected through ontology-driven weighted annotations on the documents, a classic vector-space model is utilized to evaluate the relevance between documents and queries. The weights are based on the frequency of occurrence of the instances in each document. Another ontology-based framework for semantic information retrieval has been proposed in [27]. With annotations based on GATE [28] as an information extraction module, metadata for documents are generated. The metadata are extracted concepts from the document text. Based on the term weighting technique $cf \cdot idf$ (concept frequency - inverted document frequency), which is similar to $tf \cdot idf$, concepts in the documents are recognized and indexed. Castells et al. [29] have presented a complete concept-based information retrieval model. In their model, first a set of root ontology classes is constructed from three main base classes: DomainConcept, Topic, and Document. Documents are annotated with concept instances from the ontology. The annotations are weighted based on an adaption of the $tf \cdot idf$ scheme, and the ranking is then achieved through computing similarity values between queries and documents through the classic vector-space approach. In [30], Tomassen et al. has presented an ontology-driven system WebOdIR, where each concept is extended by associating it

with a vector of key phrases describing the concept. The system starts from ranking concepts in the ontology according to ontology relevance. It then generates a query for each concept based on relations with other concepts. After submitting the queries to the underlying search engine, a set of documents for each concept is retrieved and then clustered.

Latent Semantic Indexing and Ontology

Latent semantic indexing (LSI) is similar to the vector space approach that presents documents/queries in the manner of vectors. Like vector space approaches, LSI relies on a term-document matrix in which each column represents a document and each row lists frequencies of a term in different documents. LSI uses the singular value decomposition (SVD) to reduce the dimensions of the term-document matrix and approximate the most important part of the original matrix.

Snasel et al. mapped LSI concepts to Wordnet [31]. Wordnet provides synonyms to expand queries: this improves recall through sacrificing precision, because it increased the number of keywords. LSI helps to retrieve the most relevant k terms/concepts from term matrix T_{mk}. The authors argued that the expansion of these k terms instead of all keywords balanced the conflicts between precision and recall.

3 Background

3.1 Semantic Web and Ontology

The Semantic Web [32], as an extension of the World Wide Web (WWW), makes the web content more machine-processable. Tim Berners-Lee, who invented the WWW in the late 1980s, introduced this special vision of the Web, in which the meaning of the content in Web documents would play a much more crucial role than it does today. To accomplish that the Semantic Web needs a more firmly structured language for machine agents to process. The crucial first step has been the adaptation of the language RDF (resource description framework) [33] as a standard for the Semantic Web.

The RDF is more a model than a language; it is designed to present information about Web resources. RDF presents statement in the form of triples, i.e., *subject-predicate-object*. RDF is applied to build information-sharing models. In the triples, the *subject* denotes the resources, the *object* denotes the properties of the subject, and the *predicate* denotes the relations between the subject and the object. The RDF provides a foundation for defining the main data structure of the Semantic Web – ontology.

The most popular definition of an ontology, in the context of the Semantic Web, is "an explicit and formal specification of a conceptualization of a domain of interest" [10]. Ontologies are more than just a vocabulary, they are sources of knowledge of a specific domain. Currently, most of ontologies are implemented in OWL (Web ontology language) which is based on RDF and designed by W3C.

The most important aspect of ontologies used for Semantic Web applications is related to identifying two ontology layers: *the ontology definition layer*, and *the ontology instance layer*. The ontology definition layer represents a framework used for establishing an ontology structure and for defining classes (concepts) existing in a given domain. The structure of an ontology is primarily based on a relation *is-a* between classes. This relation represents a *subClassOf* connection between a superclass and a subclass. In such a way, a hierarchy of classes is built.

The ontology definition contains descriptions of all classes of the ontology. The classes are defined using datatype properties and object properties. Both property types provide an accurate

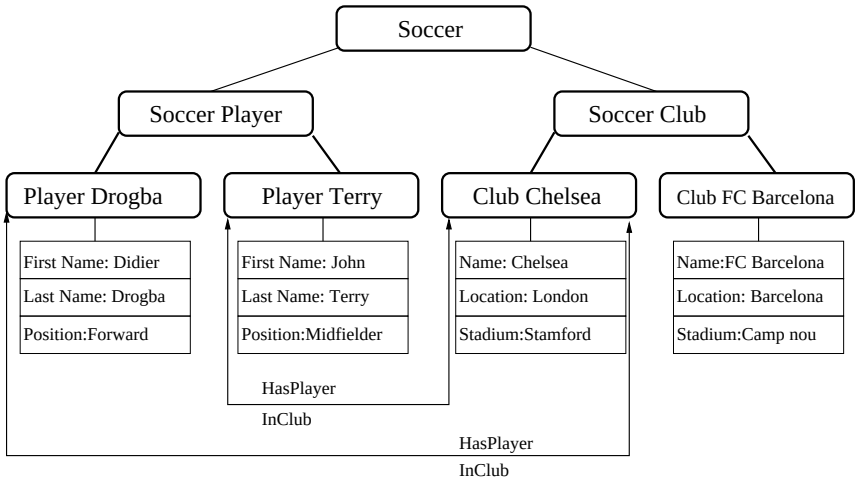


Figure 1: Ontology Snippet

and complete description of a class, as described below:

- The *datatype property* focuses on describing features of a class; datatype properties can be expressed as values of data types such as boolean, float, integer, string, and many more (for example, byte, date, decimal, time);
- The *object property* defines other than *is-a* relations among classes (nodes); these relations follow the notion of the RDF that is based on the triple subject-predicate-object, where: *subject* identifies what object the triple is describing; *predicate* defines the piece of data in the object a value is given to; and *object* is the actual value of the property; for example, in the triple John likes books, John is the subject, likes is the predicate and books is the object.

Both types of property are important for defining ontologies. The possibility of defining class properties and relations between classes creates a versatile framework capable of expressing complex situations with sophisticated classes and the multiple different kinds of relationships existing among them.

Once an ontology definition is constructed, its instances, called individuals, can be built. The properties of classes are filled out: real data values are assigned to datatype properties, and links to instances of other classes (individuals) are assigned to object properties.

A fragment of an ontology is presented in Fig. 1. It contains three classes: *Soccer*, *Soccer_ Club* and *Soccer_ Club*. Both *Soccer_ Club* and *Soccer_ Player* are *subClassOf* of the class *Soccer*. Class *Soccer_ Club* has two individuals: *Club_ Chelsea* and *Club_ FC_ Barcelona*; while Class *Soccer_ Player* has two players as individuals: *Player_ Drogba* and *Player_ Terry*.

Class *Soccer_ Player* is defined by three data properties and one object property, as presented in Fig. 1. The three data properties are *First_ Name*, *Last_ Name* and *Position*. The one object property is *InClub*. Class *Soccer_ Club* is also defined by three data properties and one object property. The three data properties are *Name*, *Stadium* and *Location*. The one object property is *HasPlayer*.

Two individuals of the class *Soccer Player* are *Player Drogba* and *Player Terry*. Individual *Player Drogba* is defined by the term "Didier" as the value of a datatype property *First Name*, by the term "Drogba" as the value of datatype property *Last Name*, by the term "Forward" as the value of datatype property *Position*, and by individual *Club Chelsea* as the value of the object property *InClub*. Similarly, individual *Player Terry* is described by "John" as *First Name*, "Terry" as *Last Name*, "Midfielder" as *Position*, and individual *Club Chelsea* as the value of *InClub*.

Individuals of the class *Soccer Club* (*Club Chelsea* and *Club FC Barcelona*) are also described by three datatype properties and an object property. The three datatype properties listed are *Name*, *Location*, and *Stadium*. The object property is *HasPlayer*, which is the inverse property[2] of the object property *InClub*. The values of the datatype properties and the object property for individuals *Club Chelsea* and *Club FC Barcelona* are shown in Fig. 1.

3.2 Ordered weighted averaging (OWA) operators

Basic principles of OWA

Aggregation of different pieces of information is a common aspect of any system that has to infer a single outcome from multiple facts. An interesting class of aggregation, ordered weighted averaging (OWA) [34] operators, is a weighted sum over ordered pieces of information.

In a formal representation, the OWA operator, defined on the unit interval I and having dimension n (n arguments), is a mapping $F_w : I^n \to I$ such that

$$F_w(a_1, ..., a_n) = \sum_{j=1}^{n} (w_j \cdot b_j) \tag{1}$$

where b_j is the jth largest of all arguments a_1, a_2, ..., a_n, and w_j is a weight such that w_j is in [0,1] and $\sum_{j=1}^{n} w_j = 1$.

If $id(j)$ is the index of the jth largest of a_i then $a_{id(j)} = b_j$ and $F_w(a_1, ..., a_n) = \sum_{j=1}^{n} (w_j \cdot a_{id(j)})$. If W is an n-dimensional vector whose jth component is w_j, and B is an n-dimensional vector whose jth component is b_j, then $F_w(a_1, a_2, ..., a_n) = W^T B$. In this formulation, W is referred to as the OWA weighing vector and B is called the ordered argument vector.

At the beginning of the 1980s, Zadeh [35] introduced the concept of linguistic quantifiers. Those quantifiers describe a proportion of objects. According to Zadeh, a person knows a vast array of terms that are used to express information about proportions. Some examples are *MOST*, *AT LEAST HALF*, *ALL* and *ABOUT ONE THIRD*. The important issue is to formally represent those quantifiers.

In the mid-1990s, Yager [36] showed how we can use a linguistic quantifier to obtain a weighing vector associated with an OWA aggregation. Yager introduced parameterized families of regular increasing monotone(RIM) quantifiers. These quantifiers are able to guide aggregation procedures by verbally expressed concepts in a description independent dimension. A RIM quantifier is a fuzzy subset Q over $I = [0, 1]$ in which for any proportion $r \in I$, $Q(r)$ indicates the degree to which r satisfies the concept indicated by the quantifier Q [36]. Assuming a RIM quantifier, we can associate with Q an OWA weighing vector W such that for $j = 1$ to n,

$$w_j = Q(\frac{j}{n}) - Q(\frac{j-1}{n}) \tag{2}$$

where n is a number of pieces of information to be aggregated. This expression indicates that the weighing vector W is a manifestation of the quantifier underlying the aggregation process. Using this expression the values of the weighing vector can be obtained directly from the expression representing the quantifier.

For example, let us take a look at the parameterized family $Q(r) = r^p$, where $p \in [0, \infty)$. Here if $p = 0$, $w_1 = 1$ and $w_j = 0$ for $j \neq 1$, and we obtain the *existential* (max) quantifier, which makes the OWA operator F closer to an *or* operator; when $p -> \infty$, $w_n = 1$ and $w_j = 0$ for $j \neq n$, and we have the quantifier $for\ all$ (min), which makes the OWA operator F closer

[2]If an object property o_1 points object A from object B, and another object o_2 points B from A, o_1 and o_2 are regarded as inverse properties.

to an *and* operator; and when $p = 1$ we have $Q(r) = r$, $w_j = \frac{1}{n}$ and we deal with the quantifier $SOME$.

OWA with Argument Importance

In the previous section we showed how a quantifier Q indicating interaction between pieces of information can be used to calculate an OWA weighing vector W. However, not all pieces of information are of the same importance. A user may desire to assign different weights (importance) to different arguments (pieces of information).

Let $m_i \in [0, 1]$ be a value associated with an argument a_1 indicating its importance. In such a case, let M be a n-dimensional importance vector $[m_1, m_2, ...m_n]$, and the weighing vector W has to be calculated based on both Q and M.

The first step is to calculate the ordered argument vector B (Eq. 1), such that b_j is the jth largest of all arguments a_1, a_2, ..., a_n. Furthermore, we assume μ_j denotes the importance weight associated with the attribute that has the jth largest value. Thus if a_3 is the largest value, then $b_1 = a_3$ and $\mu_1 = m_3$. The next step is to calculate the OWA weighing vector W using a modified version of Eq.(2),

$$w_j = Q(\frac{X_j}{T}) - Q(\frac{X_{j-1}}{T}) \tag{3}$$

where $X_j = \sum_{k=1}^{j} \mu_k$ and $T = X_n = \sum_{k=1}^{n} \mu_k$.

So, X_j is a sum of the importance of the jth most satisfied arguments, and T is the sum of all importance. When all arguments have the same importance, Eq. (3) simplifies to Eq. (2).

3.3 Hierarchy of concepts (HofC)

The idea of representing concepts as a hierarchy was introduced by Yager [37], who represented concepts with atomic attributes, words or other concepts. Using this method, a tree-like structure is established where each vertex is a concept, and terminal vertices (leaves) are attributes. The edges of the hierarchy of concepts (HofC) represent relationships that help to define concepts with other concepts and/or attributes. These edges (connections) are of significant importance to the HofC. If we assume that concept C_1 is defined by two other concepts C_2 and C_3, then the hierarchy will have two edges connecting C_2 with C_1, and C_3 with C_1. The concept C_1 is called a superconcept, and C_2 and C_3 are subconcepts. This also means that *activation* of concepts C_2 and C_3 leads to activation of C_1.

The HofC introduces a very important element, the activation of a superconcept by active sub-concepts is fully controlled by a user. There are two controlling components: importance vector M and linguistic quantifier Q. The vector M indicates the significance of each subconcept in defining a superconcept. In other words, M determines the weight of each participating subconcepts in identifying an activation level of a superconcept. The linguistic quantifier Q guides the aggregation of subconcept activations. Both M and Q determine how activation levels of subconcepts should be combined using the OWA operator.

A simple example of HofC is shown in the Fig. 2. According to the hierarchy, *conceptA* is defined as:

$$conceptA = (conceptB, conceptC, M_A, Q_A)$$

This means that *conceptA* is defined by *conceptB* and *conceptC*. M_A determines the importance of both of *conceptB* and *conceptC* in defining *conceptA*. In this case, it is a two-value vector $M_A = [M_{A-B}, M_{A-C}]$ that implies the importance of activations of both subconcepts during calculation of the activation level of the *conceptA*. The quantifier Q_A can be of any

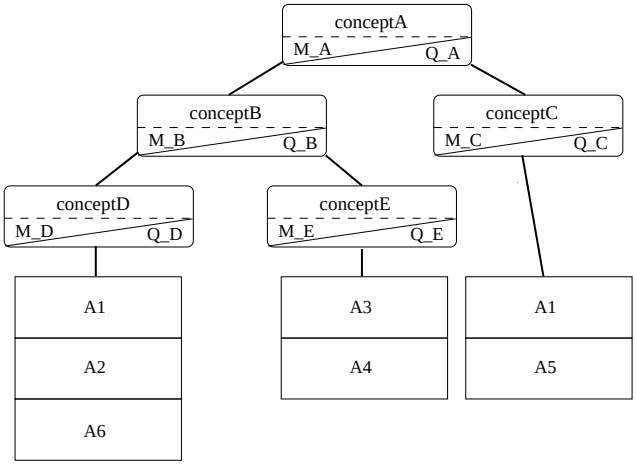


Figure 2: Simple HofC

type, for example, *most* or *some*, and identifies a mechanism of combining activation levels of subconcepts. The rest of the concepts are defined in the following way,

$$conceptB = (conceptD, conceptE, M_B, Q_B)$$
$$conceptD = (A_1, A_2, A_6, M_D, Q_D)$$
$$conceptE = (A_3, A_4, M_E, Q_E)$$
$$conceptC = (A_1, A_5, M_C, Q_C)$$

As we can see, *conceptD*, *conceptE*, and *conceptC* are defined by attributes only. Activation levels of these concepts are calculated by aggregating activations of attributes. Activation of an attribute means that the attribute is present, for example, in a Web page. The aggregation process for each concept is controlled via the M and Q associated with that concept. For *conceptD*, aggregation of activations of attributes A_1, A_2, and A_6 is performed by the OWA operators with a weighing vector determined by M_D and Q_D.

4 Concept Identification Process: Overview

A concept-based information retrieval finds relevant documents based not on simple keyword matching but on concepts that are associated with terms from users' sets of keywords. In order to accomplish that the process of concept identification is composed of the following stages:

- firstly, a user's set of keywords – called hereafter **See**d **K**eywords for **Con**cept Identification: **SeeKCon** – is translated into a structure of concepts; this is done by "mapping" keywords from SeeKCon to definitions of concepts, and constructing an equivalent Hierarchy of Concepts – HofC;
- secondly, the constructed HofC is expanded using knowledge extracted from an Ontology-based Knowledge Base – *ObKB*; SeeKCons provided by users, who usually are not experts, contains simple terms; those terms are "linked" to definitions of concepts included in ontology; the terms, other concepts, and relations associated with those definitions are used to enhance the terms that are included in the equivalent HofC; this process leads to the expended HofC;

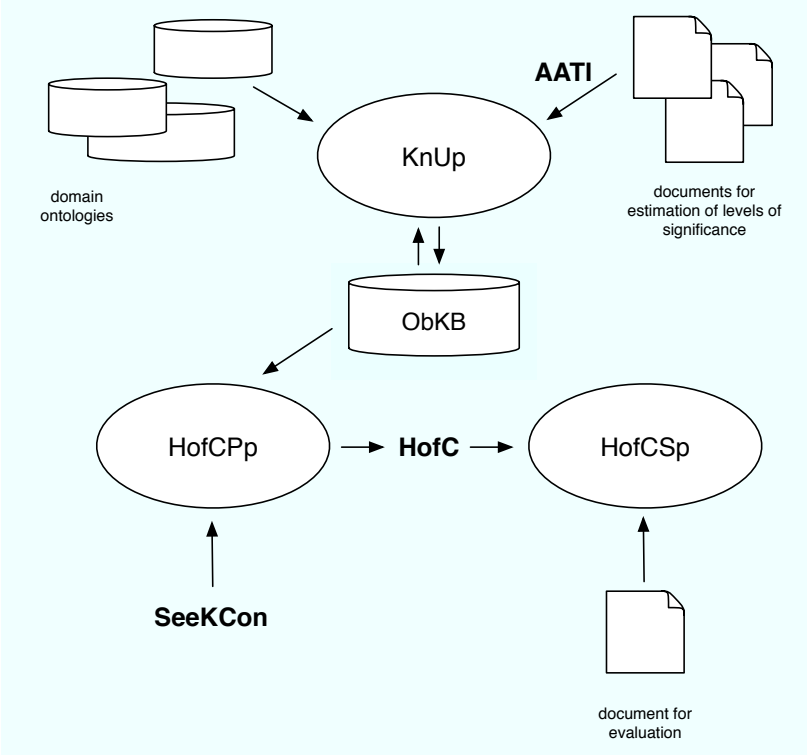


Figure 3: Overview of the Proposed Approach for Concept Identification

- finally, documents are checked how well they satisfy the extended HofC; for an HofC concept defined only by terms (no subconcepts), satisfaction of the concept is decided by the presence of those terms in a document; for an HofC concept defined by a list of terms and subconcepts, satisfaction of this concept is decided by the presence of those terms and the satisfactions of its subconcepts; in the end, the satisfaction of the extended HofC – and indirectly SeekCon – is determined by the satisfactions of concepts defined in this HofC.

An essential part of the proposed approach is a knowledge base containing definitions of concepts. These definitions are composed of (1) terms that describe these concepts together with their importance; (2) related concepts and their importance; (3) relations between terms and concepts. This knowledge base is built around ontology representing domains of interest. Overall, the concept identification process contains a number of activities: a knowledge updating process *KnUp*, a HofC preparation process *HofCPp*, and a HofC satisfaction process *HofCSp*. *KnUP* works with a knowledge base to provide extra definitions of concepts and updates of single keywords importance in defining concepts. *HofCPp* translates users' sets of keywords into HofCs, and expands them with knowledge from a domain knowledge base. *HofCSp* estimated levels of activation of concepts in documents what leads to identification of concepts.

Fig. 3 presents an overview of the proposed concept identification process. Component *KnUp* builds and manages the *ObKB*, the knowledge base. The importance values for terms or concepts in the *ObKB* are obtained by the AATI scheme, as shown in Fig. 3. The *document repository* contains "unknown" to the AATI documents. That is, there is no need for human experts to work on these documents to generate such knowledge as contained concepts, categories they belong to and so on. Therefore, the process of updating the *ObKB* does not stop. This is an outstanding advantage of the approach because no extra human labor is needed to build sample documents

and the knowledge in the *ObKB* can be kept up to date.

HofCPp translates users' SeeKCons into equivalent HofCs. It enriches these HofCs with knowledge stored in the *ObKB*. HofCs provide lists of concepts to be identified and their definitions. The definitions of concepts, also known as the knowledge of concepts, include subconcepts, terms, their importance and their relations. As the output of this process, expanded HofCs are utilized for evaluation.

HoCSp determines how HofCs are activated by documents. Documents are semantically annotated first. The annotated terms are pieces of information that satisfy concepts from the HofC. OWA is integrated with the HofC to evaluate the satisfactions of queries by texts in documents. This process is mapped as the aggregation of the activations of the concepts from the HofC.

The proposed process successfully implements concept identification. A term, as a piece of atom information, is not a binary string, but is related to a concept that is a meaningful entity defined in the *ObKB* or in an HofC. That is, it has semantics. This is accomplished based on semantics instead of a simple presence. The main techniques, such as an ontology-based knowledge base (the *ObKB*), an AATI scheme, a HofC format, and OWA operators for aggregation, are applied for this purpose.

- An ontology provides a specification of a conceptualization. That is, it provides lists of concepts, terms and their relations in a domain.
- An AATI scheme assigns importance to terms and in turn to concepts in the *ObKB* by "blindly" reading of documents.
- Users' SeeKCons are organized into concepts in hierarchical structures (HofC).
- OWA operators aggregate the satisfactions of concepts in a HofC to evaluate the satisfaction of the whole HofC. The HofC accepts different linguistic quantifiers that make the approach flexible in defining concepts and representing different user interests.

5 Ontologies and Ontology-based Knowledge Base

In order to identify high-level concepts we need to find out how many terms and subconcepts from concept definitions are present in a document. Concepts which have their definitions "highly activated" are concepts identified in the document. The process of concept identification requires two essential parts:

1) *definitions of concepts* that contain lists of related concepts, subconcepts and terms, as well as relations between them: here, we use ontology as source of such information;
2) *levels of importance of concepts and terms*: they represents levels of contribution of terms/-concepts toward concept activation levels; to address that issue we have developed Adaptive Assignment of Term Importance (AATI) schema.

There are two very important parts of *KnUp*: one that builds ontology-based knowledge base – *ObKB* – from existing domain ontologies, and the other one that updates *ObKB*. The process of updating the *ObKB* is accomplished by the new self-adaptive scheme, AATI (Adaptive Assignment of Term Importance), which assigns importance values to terms/concepts in the *ObKB*, and continuously updates them.

5.1 Ontology-based Knowledge Base (ObKB)

In an ontology, a *class* is described with *properties*. There are two types of properties: datatype property, and object property (Section 3.1). In such a case, a concrete piece of infor-

mation – called an *individual* – is an instance of a class and is created by assigning values to the *properties*. The ontology as defined above does not contain information required for concept identification in a suitable format. There are few main reasons for that:

1. a HofC contains concepts, while an ontology contains classes and individuals; it is difficult to expand HofCs directly from ontology without a transformation process;
2. HofC connections and ontology connections have different meanings;
3. an ontology does not contain importance (represented by vector M), and linguistic quantifiers (represented by Q) which are necessary for OWA operators to determine levels of activation of concepts contained in HofCs (section 3.2).

Ontology vs. HofC: Component Aspect

In the Semantic Web definition of ontology, classes are abstract and individuals represent concrete information. That is, class defines properties that have no concrete values; while individuals are defined by values assigned to the properties. Unlike an ontology that is designed to store a large amount of organized knowledge, a HofC represents a list of specific information relevant to a high level concept. If a class in an ontology has no individuals, i.e., no concrete information associated with it, it can be provided later by others. But it is not reasonable that a HofC should contain a vertex without a concrete value associated with it. Such a concept is useless because there is no way to identify it in a document. Therefore, both classes as well as individuals from domain ontologies are translated into concepts in the *ObKB*. The following formats and operations pertain here:

- *concepts* are used in the HofC and the *ObKB*; *classes* and *individuals* are used in the ontology;
- *attributes* are used in the HofC and the *ObKB*; *properties* are used in ontology;
- the attributes in the HofC and the *ObKB* are called *terms*, which can be found in documents.
- OWA operators are applied to aggregate activation levels of terms and concepts included in HofCs; OWA operators require M (importance) and Q (linguistic quantifiers)[3].

Ontology vs. HofC: Conncection Aspect

Another important difference between ontologies and HofCs is related to connections between components existing in both structures.

The *ObKB* has the same connections as ontology. These connections have different meanings. For example, let us assume there is a *ObKB* built based on the ontology shown in Fig. 1. The connection between the concept *Player Drogba* and the concept *Soccer Player* is different from the connection between the concept *Player Drogba* and the concept *Club Chelsea*. The former connection means "is-a" or "has-a." That is, *Player Drogba* is-a *Soccer Player* or *Soccer Player* has-a *Player Drogba*. The latter connection means "InClub" or "HasPlayer". That is, *Player Drogba* InClub *Club Chelsea* or *Club Chelsea* HasPlayer *Player Drogba*.

The connections in HofCs represent relative levels in the hierarchy structure, that is, one concept is a subconcept (superconcept) of another concept. HofCs are built to represent SeeKCons where subconcepts contribute to defining their related superconcepts. The connection may be meaningless, because the user can connect any concepts to express his/her interests. For example, let us assume the knowledge in the ontology shown in Fig. 1 is correct. A user can create

[3]Details about M and Q are in section 3.2 and section 3.3

his/her own HofC which has player *Player Drogba* as the superconcept, and *Club FC Barcelona* as the subcocnept if he/she assumes that Player Drogba will transfer to Club FC Barcelona and he/she wants to check if there is any rumour about it. In this case, the satisfaction of *Club FC Barcelona* contributes to the satisfaction of *Player Drogba*, and in turn to the satisfaction of the whole HofC, though it seems the connection should not exist.

In general, subconcepts (superconcepts) are used to describe two connected concepts in a HofC; while the term *related-concepts* is used to describe two connected concepts in the *ObKB*. Overall, we define three types of connections.

1. $TypeI$ connection connects concepts that are parts of users' SeeKCons; in other words the concepts provided by users are related to each other through $TypeI$ connections; they are the strongest type of connections;
2. $TypeII$ connection connects concepts defined in the domain ontology; these connections are "is-a" and "has-a" relations; the first stage of expanding an equivalent HofC is based on this type of connection (Section 6.2);
3. $TypeIII$ connection also connects concepts defined in an ontology; however these connections represent all types of relations except "is-a" or "has-a" relation; this is the weakest connection; the second stage of expanding an equivalent HofC is based on this type of connection (Section 6.2).

Mapping from Ontology to ObKB

In a nutshell, the process of building the *ObKB* from domain ontologies consists of a few steps which transforms ontology specific elements into components suitable for construction of HofC. Those steps are:

1. a class from ontology is translated into a concept, however the definitions of properties are not translated;
2. an individual from ontology is translated into a concept, and
   - the values of object properties of this individual are translated into concepts attached to the concept created based on this individual;
   - the values of datatype properties of this individual are translated into attributes attached to the concept created based on this individual;
3. a default value of Q is added to the created concept; the default Q is $MOST$;
4. the default value of M is set as zero for each concept and term.

The *ObKB* is used to expand equivalent HofCs created based on SeeKCons provided by users.

5.2 Estimating Importance Levels

The *ObKB* is a model of a given domain. It contains concepts, terms, and relations that are specific for this domain. The AATI scheme integrated with the *ObKB*, determines importance values that represent levels of contributions of these concepts and terms towards concept activation.

In [39], the AATI was studied and validated through experiments. The AATI scheme is implemented based on the power iteration [38], which is used to find eigenvectors of a matrix in linear algebra. The main steps of AATI implementation are as follows:

1. translate an ontology into the *ObKB*;

2. take a new document;
3. parse the document and annotate it with concept names an terms from the *ObKB*;
4. for each term do one of the following:
   (a) *if Term Weight* is not zero (it means the term has already appeared in documents), take this as its new *Term Weight*;
   (b) *if Term Weight* is zero (the term has not been found in any documents), randomly generate a number between 0 and 1, and assign it as its *Term Weight*;
5. calculate the *Page Value* of this document;
6. update the *Term Weights* of those terms found in the document;
7. normalize *Term Weights* across all terms (make the sum of *Term Weights* equal to 1);
8. if there are no more documents, *STOP*; otherwise go back to Step 2.

The AATI continuously updates *Term Weights* based on available documents. Intuitively, we can say that the *Page Value* is high when terms occurring in a document have high *Term Weights*. At the same time, we can state that the *Term Weight* of a term is high when the *Page Values* of documents that contain the term are high.

5.3 ObKB-based Annotation of Documents

We have developed a Java annotation module based on the UIMA (Unstructured Information Management Architecture) library created by IBM [40], which adds extra information to the texts of documents.

Annotation is the basis for concept identification processes. Terms, which are literal values of the datatype properties – attributes in the *ObKB* – are used to identify concepts in texts. All occurrences of *ObKB* terms in a document are identified, and annotation information is added to them. This information includes its position (begin, end) in a document, the concept it belongs to, and the attribute it belongs to. Annotation connects the *ObKB* with documents: the frequency of a term presented in a document is stored and used later for determination how well the document satisfies the concept, to which the term is attached in the *ObKB*.

Fig.4 shows how a document is annotated. For example, the term "Chelsea" from *ObKB* is the attribute *Name* attached to the concept *Club Chelsea*. The annotation module finds that this term is present twice in the document. So, for the *ObKB* and the annotation module, a string "Chelsea," which to most computer agents is meaningless (nothing but a binary string), has its own semantics/meaning: it is the name of a soccer club.

6 HofC Preparation Process

The *HofCPp* preprocesses SeeKCons, builds HofCs from them, and expands these HofCs with knowledge from the *ObKB*.

The preprocessing of SeeKCons removes stop words and stems words. Stop words include articles such as "the" and "a" and prepositions such as "in" and "from." The stemming process removes suffixes and prefixes reducing words to their stems. For example, the term "extended" can be stemmed to "extend."

The process of building equivalent HofCs translates SeeKCons into concept structures as explained in Section 6.1. Next, knowledge is retrieved from the *ObKB* to enhance the definitions of concepts. The details are presented in Section 6.2.

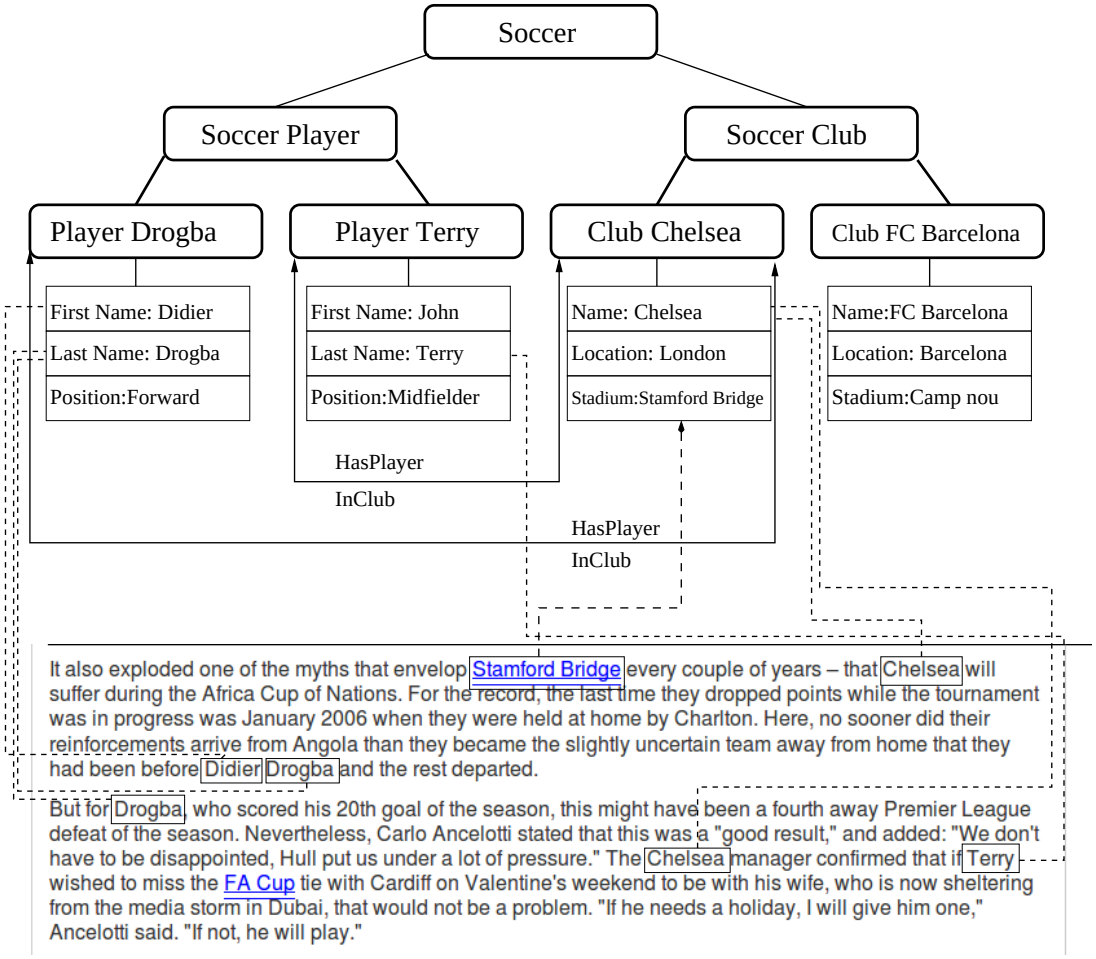


Figure 4: Document Annotation: Example

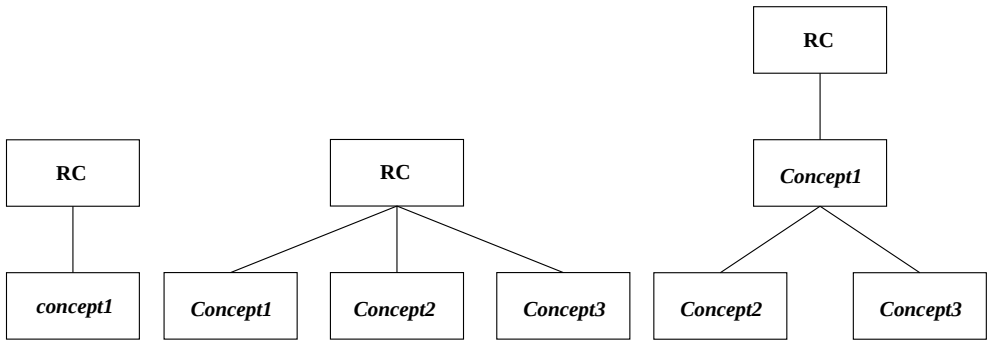
6.1 Representation of Users' Sets of Keywords as HofCs

A SeeKCon is a statement representing the interest of the user. The user provides keywords that describe objects that the user desires to identify in a document. The keywords are represented as a flat structure, that is, there is no indication that the keywords are related to each other, or that some keywords depend on others. This is a simplification suitable for automatic processing but it is not a realistic representation of a human-like way of finding concepts. For a human, all keywords are interconnected. They constitute a network of words representing concepts. The activation of a single word initiates activation of a concept and related concepts.

A SeeKCon is represented by a HofC – a tree-like structure built of concepts (vertices) and terms (terminal vertices/leaves) (Section 3.3). This structured form of a SeeKCon, that resembles a network of words and concepts, provides a more intuitive way of expressing things the user is looking for. A document that satisfies an equivalent HofC contains a high-level concept related to this HofC. The satisfaction of the HofC is a result of an aggregation of satisfactions performed at each vertex of HofC using OWA, and associated with it the linguistic quantifier Q, and the vector M of importance values.

As presented in Section 3.2, a linguistic quantifier Q associated with a vertex is used at the time of the aggregation of satisfactions of terms and other concepts (subconcepts) attached to this vertex. The possible quantifiers are *SOME*, *MORE*, etc. The notion of linguistic quantifiers plays an important role in representing different ways of combining satisfactions.

A vector M (Section 3.2) represents the importance of each term and concept that is taken into consideration during an aggregation process. Not all terms and concepts contribute uniformly to the activation of a concept associated with a vertex of HofC.



(a) Keyword (b) Multiple Keywords (c) Structured Keywords

Figure 5: Sample Root Concepts (RCs)

A SeeKCon represents a high-level concept to be identified in documents. The goal is to find out if a given document contained terms and subconcepts that are part of the equivalent HofC. This translates into a process of estimating a level of activation (or satisfaction) of the HofC. In order to simplify this process, a special concept, called *Root Concept (RC)*, is automatically added to each equivalent HofC. In this case, the activation of the HofC is the same as the activation of the top level concept of the equivalent HofC.

Some possible structures of HofCs are presented in Fig 5. In Fig. 5a, a SeeKCon contains one single concept; in Fig. 5b, a SeeKOnt contains multiple concepts in a flat structure; and in Fig. 5c, a SeeKCon contains a few structured concepts. Activation of these HofCs is simplified to the activation of a single root concept (RC), regardless the complexity of HofCs.

6.2 Building and Expanding HofC

The process of identifying high-level concepts in a document depends on the ability to determine if HofC concepts can be inferred based on the texts of the document. This depends on the contents of the HofC itself – the more terms and concepts are used to build the HofC, the higher the chance of proper evaluation of presence of a high-level concept in a document. We do not expect users to provide comprehensive SeeKCons containing a large number of terms and concepts. The model accepts conventional keyword-based SeeKCons that are automatically translated into HofCs (Section 6.1). That is, the keywords are translated into concepts with a specific hierarchy representing how those concepts are related to each other. The keywords from SeeKCon are regarded as connected with a *typeI* connection.

The names of concepts from an equivalent HofC (the same as keywords provided by the user) are used to retrieve knowledge of equivalent concepts from the *ObKB*. The knowledge includes subconcepts, attributes, linguistic quantifiers (Q), and importance vectors (M). The expansion process is as follows.

for each *concept* from the HofC **do**
search *ObKB* for a concept equivalent to the *concept*
if an equivalent concept is found **then**
1. copy all concepts from the *ObKB* connected via connection $TypeII$ to the found concept as subconcepts of the *concept*

2. copy all attributes of all just copied concepts from *ObKB* as attributes of the subconcepts
3. copy all concepts, and their attributes, connected to the concept found in *ObKB* via $TypeIII$ connection

else
add a string-type attribute "NAME" to the *concept*; the value of this attribute is the name of the *concept*
end if
end for

A HofC includes concepts and attributes arranged in a hierarchy. Identification of a concept in the *ObKB* through a keyword means finding the matching keyword among the concept names or concept attributes in the knowledge base. Once an equivalent concept is found, all knowledge regarding it present in the *ObKB* is used to enhance the HofC.

It is also possible that the user provides other linguistic quantifiers Q different from the default quantifier contained in the *ObKB*. Different linguistic quantifiers mean different ways of aggregation of activation levels (Section 3.2).

7 Identification of Concepts: HofC Satisfaction Process

HofC Satisfaction Process (*HofCSp*) accepts expanded HofC (output from *HofCPp*), and identifies concepts in documents.

An extended HofC contains not only concepts (user's keywords) themselves but also concepts and attributes from the definitions of keyword-based concepts. Those "new" concepts and attributes can be identified in documents.

The evaluation of relevance is mapped as the "activation" of the HofC based on OWA (Section 3.2). The more of the HofC is activated, the higher the chance that the document contains a high-level concept represented by HofC.

7.1 Satisfaction, Importance and Aggregation weights

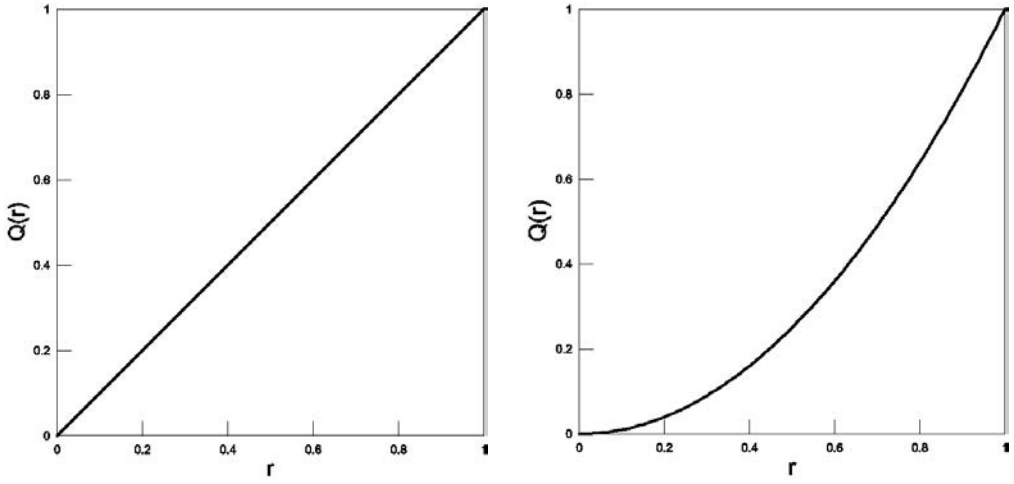
Before explaining how *HofCSp* works, three terms used here: satisfaction, importance, and aggregation weights, need to be defined

- *Satisfaction* represents how well a *criterion* in the HofC is satisfied by the text from a document. A *criterion* can be an attribute, a concept or even a complete HofC. The satisfaction of the root concept (RC) (Section 6.1) is equivalent to the satisfaction of the whole HofC.
- *Importance* represents how important a criterion is. With the AATI scheme, each entity (term or concept) in the *ObKB* includes a measure of importance.
- *Aggregation weight* is the weight of a criterion calculated by the OWA operator (Section 3.2). Aggregation weight depends not only on the importance of the criterion but also on the linguistic quantifier. In Section 7.2, the use of linguistic quantifiers and their importance in calculating aggregation weights is explained.

7.2 Linguistic Quantifier

The concept of linguistic quantifiers was introduced by Zadeh [35] in the early 1980s. Some examples are *ALL*, *MOST*, *AT LEAST HALF* and *ABOUT ONE THIRD*. To formally rep-

resent those quantifiers, Zadeh suggested using a fuzzy subset $Q(r)$ as a linguistic expression corresponding to a quantifier that indicates the degree to satisfy the concept for any proportion $r \in [0, 1]$.



(a) *some* (b) *most*

Figure 6: Linguistic quantifiers

Yager [36] used the linguistic expression $Q(r)$ to obtain a weighting vector W associated with an OWA aggregation. These quantifiers were then able to guide aggregation procedures by verbally expressed concepts in a description independent dimension.

In the presented work, two linguistic quantifiers are considered:

- $Q(r) = r$ represents the linguistic quantifier *some*, Fig. 6a. The equation means that the aggregation weight of a criterion depends only on its importance value. That is, the relations between concepts do not change their aggregation weights.
- $Q(r) = r^2$ represents linguistic quantifier *most*, Fig. 6b. Using this equation, the aggregation weight of a criterion tends to be large when it is less satisfied (more details are in section 3.2).

7.3 Satisfaction of Concept with Attributes Only

A criterion for a single concept can be an attribute or a subconcept. The simplest situation is when a concept has only attributes attached to it.

The literal values of attributes are called *terms*. The presence (more precisely, frequencies) of a term or terms in a document decides how well the criterion (attribute) is satisfied. Mathematically, the satisfaction of a term s_i is expressed with the following formula:

$$s_i = e^{-\frac{1}{freq_i}} \tag{4}$$

where $freq_i$ is the frequency of the term in the document. The reasons for using Eq. 4 are:

1. It is a monotonically increasing function, so when its frequency increases, the satisfaction increases.
2. The function "saturates" with increasing values of the argument, so if a given term occurs many times in a document, its satisfaction does not overshadow the satisfactions values of other terms.

Each criterion has an assigned value of importance. Assuming there are n different datatype attributes attached to a single concept C, the importance vector M contains n elements, where each element represents the importance of a criterion. The satisfaction vector S also contains n elements, where each element represent the satisfaction level calculated for a criterion based on a given document. In such case, the ordered satisfaction vector S^{prime} (Eq. 1) is formed based on S, in which s_j^{prime} is the jth largest of all satisfactions s_1, s_2, ..., s_n. Furthermore, we assume μ_j denotes the importance weight associated with the attribute that has the jth largest value. Thus if s_3 is the largest value, then $s_1^{prime} = s_3$ and $\mu_1 = m_3$. Based on this, the OWA weighing vector W is obtained by $Q(r)$ in Eq. 5,

$$w_j = Q(\frac{X_j}{T}) - Q(\frac{X_{j-1}}{T}) \tag{5}$$

where $X_j = \sum_{k=1}^{j} \mu_k$ and $T = X_n = \sum_{k=1}^{n} \mu_k$. That is, X_j is the sum of the importance weights of the jth most satisfied arguments, and T is the sum of all importance weights.

So, the satisfaction of concept S_c is:

$$S_c = \sum_{k=1}^{n} w_k \cdot s_k^{prime} \tag{6}$$

7.4 Satisfactions of Concept with Attributes and Subconcepts

As mentioned earlier, a criterion can be an attribute or a subconcept. The satisfaction of a subconcept is the satisfaction of a concept. The calculations of the satisfaction of a criterion based on terms (attributes) and the calculation of the satisfaction of a criterion based on concepts (subconcepts) are different. The calculation of the satisfaction of a criterion based on terms is quite simple, as shown in Eq. 4, while the calculation of the satisfaction of a criterion based on concepts includes more complicated aggregation. When both attributes and subconcepts are attached to the concept, we aggregate separately satisfactions of attributes and satisfactions of subconcepts, and then aggregate the results of both aggregations.

For example, for the *conceptD* in Fig. 7 its satisfaction is calculated using Eq. 6 using frequencies of terms. In the case of the *conceptA* in Fig. 7, its satisfaction is calculated in two steps: firstly, the satisfaction of the $TermA1$ is determined as well as satisfactions of the *conceptB* and the *conceptC*; secondly, all satisfactions are aggregated using OWA with M_A and Q_A.

7.5 Satisfaction of HofC

Activation of subconcepts the HofC is propagated upward as the process to calculate satisfaction of the HofC continues. That is, calculation of the satisfaction of the HofC starts from the attributes/concepts at the lowest level and ends with the concepts at the highest level. The presence of terms/attributes (for example, $TermD1, TermD2, TermC1$, and $TermC2$, see Fig. 7) in a document contributes to the satisfaction of the corresponding concepts ($ConceptD$ and $ConceptC$, respectively, Fig. 7). The aggregation of the satisfactions of those concepts and other terms contributes to the satisfaction of the higher-level concepts (for example, $TermB1$ and $ConceptD$ lead to activation of $ConceptB$, Fig. 7). The process is repeated until the satisfaction of the concept on the top level is calculated.

The structure of HofC determines which terms/attributes and concepts contribute to the satisfactions of which concepts – satisfaction of a single concept is calculated based on the satisfactions of all terms and concepts that are attached (from the bottom) to it. The aggregation of satisfactions is performed at each concept of HofC using OWA (Eq. 6).

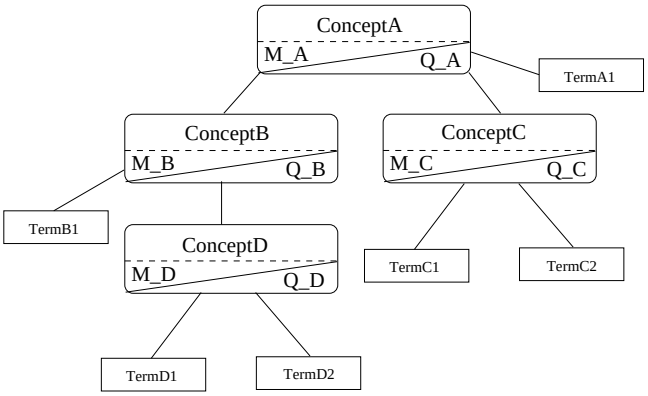


Figure 7: Enhanced HofC

8 Case Study

In order to illustrate the proposed process for concept identification we present a simple case study where two SeeKCons: A:Chelsea and B:Drogba;Terry;Chelsea, are "checked" against two documents Doc.1 and Doc.2. The process is performed with two linguistic quantifiers *MOST* ($Q(r) = r^2$) and *SOME* ($Q(r) = r$). The frequencies of terms in Doc.1 and Doc.2 are listed in Table 1. Roughly, Doc.1 contains more information about *Concept Chelsea* and Doc.2 contains more information about *Player Drogba* and *Player Terry*.

Table 1: Term frequencies in documents

	Doc.1	**Doc.2**
Concept Chelsea		
Name:Chelsea	6	0
Location:London	2	1
Stadium:Stamford	2	1
Concept Player Drogba		
First Name:Didier	0	2
Last Name:Drogba	1	2
Position:Forward	0	1
Concept Player Terry		
First Name:John	0	2
Last Name:Terry	1	3
Position:Midfielder	0	0

8.1 Building and Expanding HofC

Both SeeKCons A and B are used to create HofCs. HofC_A is built from the keyword "Chelsea", Fig. 8. The keyword "Chelsea" itself is a piece of meaningless text for the machine. However, *KnUp* tries to figure out the concept embedded in this keyword. It finds the concept *Club Chelsea* in the *ObKB* because the term "Chelsea" is the value of its attribute *Name*. Then definitions of the concept in the knowledge base are added into the SeeKCon including the

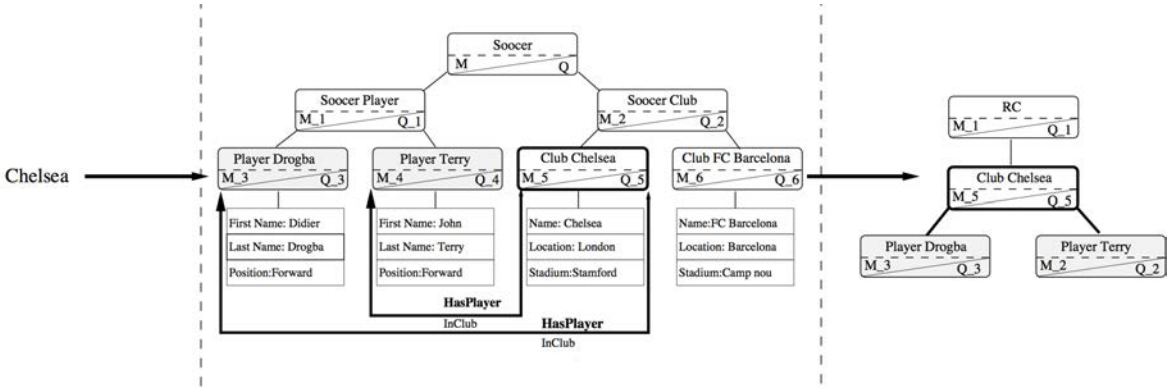


Figure 8: Construction of HofC_A

values of its datatype attributes such as "London" (attribute *Location*) and "Stamford"(attribute *Stadium*), and the values of the object attribute *HasPlayer* associated with the two concepts: concept *Player Drogba* and *Player Terry*. The connection between concept *Club Chelsea* and concept *Player Drogba* and the connection between concept *Club Chelsea* and concept *Player Terry* are all $TypeIII$ connections (Section 6.1 for more details). For simplicity, in this chapter we treat $TypeI$, $TypeII$ and $TypeIII$ connections equally in calculation.

Similarly, HofC_B is built from keywords "Drogba," "Terry," and " Chelsea", Fig. 9. Since there is no indication about their relations, the process *HofCPp* places the three concepts in a flat (equal level) structure. The connections in this HofC are $TypeI$.

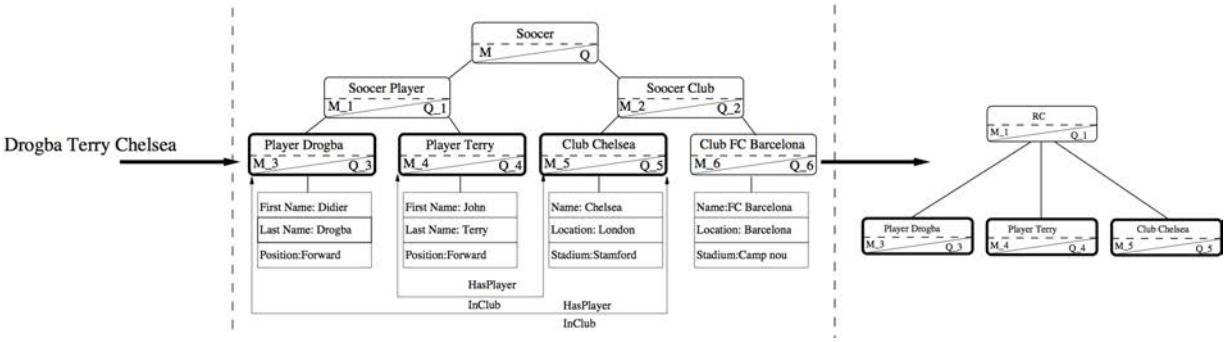


Figure 9: Construction of HofC_B

8.2 Importance of Terms

The importance (TW) of terms is assigned by the AATI. The values are displayed in the table below:

Concept Player Drogba			
Term	Didier	Drogba	Forward
Importance	0.06	0.11	0.02
Concept Player Terry			
Term	John	Terry	Midfielder
Importance	0.03	0.07	0.06
Concept Club Chelsea			
Term	Chelsea	London	Stamford
Importance	0.15	0.02	0.08

8.3 Evaluation

In total, there are four cases using the linguistic quantifiers *MOST* and *SOME*: HofC_A(most), HofC_A(some), and HofC_B(most), HofC_B(some). Cases HofC_(most) and HofC_A(some) utilize HofC_A; cases HofC_B(most) and HofC_B(some) utilize HofC_B. Table 2 shows the design.

Table 2: Case study design

Case	*Linguistic Quantifier*
HofC_A(most)	most
HofC_A(some)	some
HofC_B(most)	most
HofC_B(some)	some

Due to space limitations we show all computational details for two of those cases: HofC_A(some) and HofC_B(most). The results for the rest of the case are given in the table in Section 8.4.

Case HofC_A(most) for Doc. 1

HofC_A, shown in Fig. 8, is generated from the keyword "Chelsea." It is composed of three concepts, which are *Club Chelsea*, *Player Drogba*, and *Player Terry*. The linguistic quantifier in this case is *MOST*.

Fig. 10 displays the aggregation orders in both cases: HofC_A(most) and HofC_A(some). The ellipses in the figure denote the aggregation processes; the numbers in the ellipses are the orders of the processes. As shown in Fig. 10, satisfactions of the concepts *Player Drogba* and *Player Terry* are calculated firstly. As subconcepts of the concept *Club Chelsea*, their satisfactions are aggregated. Then satisfactions of the terms in concept *Club Chelsea* are calculated and aggregate with the satisfactions of the aggregated subconcepts, which is the satisfaction of the whole HofC.

Player Drogba has no subconcept, so its satisfaction is decided by attached terms. The term satisfaction is calculated based on Eq. (4). We calculate the ordered satisfactions of the terms in *Player Drogba*:

Concept Player Drogba	satisfaction	importance
term: Drogba	0.3679	0.11
term: Didier	0	0.06
term: Forward	0	0.02

Because the linguistic quantifier is *MOST*, which is defined by $Q(r) = r^2$. For the concept *Player Drogba*, the sum of the importance of all attached terms is $0.11 + 0.06 + 0.02 = 0.19$. Based on OWA, the weights of its criteria/terms are:

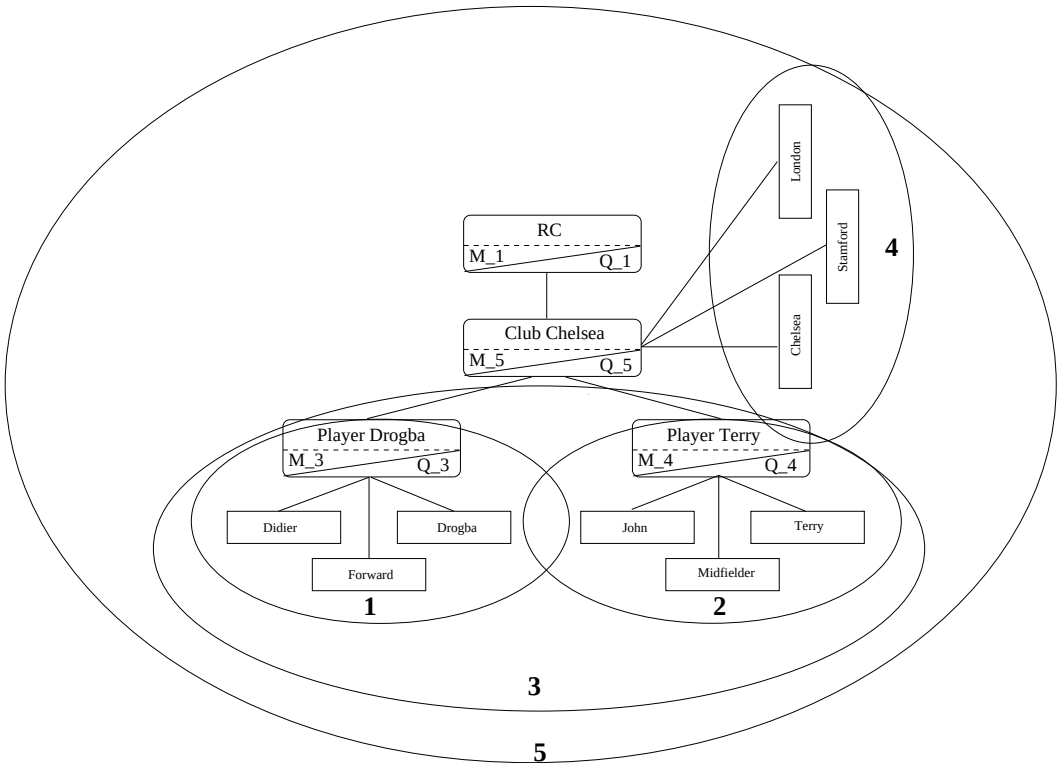


Figure 10: HofC_A: Satisfaction Calculation Process

term Drogba:

$$w_{tDrogba} = Q(\frac{0.11}{0.19}) - Q(\frac{0}{0.19}) = 0.3352$$

term Didier:

$$w_{tDidier} = Q(\frac{0.17}{0.19}) - Q(\frac{0.11}{0.19}) = 0.4654$$

term Forward:

$$w_{tForward} = Q(\frac{0.19}{0.19}) - Q(\frac{0.17}{0.19}) = 0.1994$$

For Doc.1, concept *Player Drogba* has its satisfaction as:

$$\begin{aligned} s_{cDrogba} &= 0.3352 \times 0.3679 + 0.4654 \times 0 + 0.1994 \times 0 \\ &= 0.1233 \end{aligned} \tag{7}$$

Then we calculate the ordered satisfactions of the terms in concept *Player Terry*:

Concept Player Terry	satisfaction	importance
term: Terry	0.3679	0.07
term: John	0	0.03
term: Midfielder	0	0.06

The sum of the importance of all attached terms in concept *Player Terry* is 0.07+0.03+0.06 = 0.16. Based on OWA, the weights of its criteria/terms are calculated:

term Terry:

$$w_{tTerry} = Q(\frac{0.07}{0.16}) - Q(\frac{0}{0.16}) = 0.1914$$

term John:

$$w_{tJohn} = Q(\frac{0.10}{0.16}) - Q(\frac{0.07}{0.16}) = 0.1992$$

term Forward:

$$w_{tForward} = Q(\frac{0.16}{0.16}) - Q(\frac{0.10}{0.16}) = 0.6094$$

For Doc.1, the concept *Player Terry* has its satisfaction as:

$$\begin{aligned} s_{cTerry} &= 0.1914 \times 0.3679 + 0.1992 \times 0 + 0.6094 \times 0 \\ &= 0.0704 \end{aligned} \tag{8}$$

Defined by the structure of the HofC_A, satisfaction of the concept *Club Chelsea* is obtained by aggregating the satisfactions of its subconcepts (concept *Player Drogba* and concept *Player Terry*) and terms("Chelsea," "London," and "Stamford"). Because the methods for calculateing the satisfaction of subconcepts and terms are different, we calculate the satisfactions separately and then combine them. The details are as follows.

The ordered satisfactions of sub-concepts in concept *Club Chelsea* are:

Concept Player Chelsea	satisfaction	importance	weights
concept: Drogba	0.1233	0.19	0.2947
concept: Terry	0.0704	0.16	0.7053

Satisfaction of the combination of sub-concepts in concept *Club Chelsea* is:

$$\begin{aligned} s^{prime}_{cChelsea} &= 0.1233 \times 0.2947 + 0.0704 \times 0.7053 \\ &= 0.0860 \end{aligned} \tag{9}$$

The ordered satisfactions of the terms in *Club Chelsea* are:

Concept Player Chelsea	satisfaction	importance	weights
term: Chelsea	0.8465	0.15	0.3600
term: London	0.6065	0.02	0.1024
term: Stamford	0.6065	0.08	0.5376

Satisfaction of the combination of the terms in *Club Chelsea* is:

$$\begin{aligned} s"_{cChelsea} &= 0.8465 \times 0.3600 + 0.6065 \times 0.1024 + 0.6065 \times 0.5376 \\ &= 0.6929 \end{aligned} \tag{10}$$

Satisfaction of *Club Chelsea* is calculated by the combination of subconcepts $s^{prime}_{cChelsea}$ and terms $s"_{cChelsea}$ as:

Concept Club Chelsea	satisfaction	importance	weights
comb. of terms	0.6929	0.25	0.1736
comb. of sub-concepts	0.0860	0.35	0.8264

Finally, when applied to Doc.1, satisfaction of the HofC_A is the same as satisfaction of concept *Club Chelsea*:

$$\begin{aligned} s_{cChelsea} &= 0.6929 \times 0.1736 + 0.0860 \times 0.8264 \\ &= 0.1914 \end{aligned} \tag{11}$$

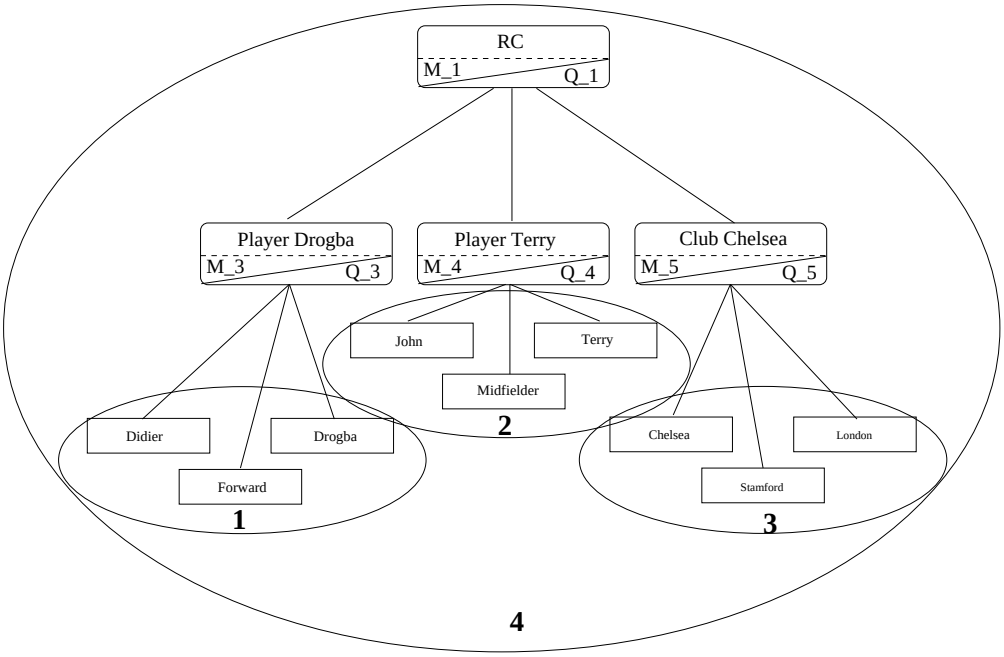


Figure 11: HofC_B: Satisfaction Calculation Process

Case HofC_B(some) for Doc. 1

If a SeeKCon "Chelsea Drogba Terry" is provided, a new HofC is created as shown in Fig. 9.

Fig. 11 displays the aggregation order for HofC_B in cases HofC_B(most) and HofC_B(some). The ellipses in the figure denote the aggregation processes; the numbers in the ellipses are the orders of the processes. As shown in Fig 11, the three single concepts (concepts *Player Drogba*, *Player Terry*, and *Club Chelsea*) are processed first. The satisfaction of the three separated concepts are aggregated as the satisfaction of the whole HofC.

First, we calculate the ordered satisfaction of the terms in *Player Drogba* as:

Concept Player Drogba	satisfaction	importance
term: Drogba	0.3679	0.11
term: Didier	0	0.06
term: Forward	0	0.02

Because the linguistic quantifier is *some*, which is defined by $Q(r) = r$. For *Player Drogba*, the sum of the importance of all attached terms is $0.11 + 0.06 + 0.02 = 0.19$. Based on OWA, the weights of its criteria/terms are:

term Drogba:

$$w_{tDrogba} = Q(\frac{0.11}{0.19}) - Q(\frac{0}{0.19}) = 0.5789$$

term Didier:

$$w_{tDidier} = Q(\frac{0.17}{0.19}) - Q(\frac{0.11}{0.19}) = 0.3158$$

term Forward:

$$w_{tForward} = Q(\frac{0.19}{0.19}) - Q(\frac{0.17}{0.19}) = 0.1053$$

For Doc.1, *concept Drogba* has its satisfaction as:

$$\begin{aligned} s_{cDrogba} &= 0.5789 \times 0.3679 + 0.3158 \times 0 + 0.1994 \times 0 \\ &= 0.2130 \end{aligned} \tag{12}$$

We calculate the ordered satisfactions of the terms in *Player Terry*:

Concept Player Terry	satisfaction	importance
term: Terry	0.3679	0.07
term: John	0	0.03
term: Midfielder	0	0.06

The sum of the importance of all attached terms in *Player Terry* is $0.07 + 0.03 + 0.06 = 0.16$. Based on OWA, the weights of its criteria/terms are:

term Terry:

$$w_{tTerry} = Q(\frac{0.07}{0.16}) - Q(\frac{0}{0.16}) = 0.4375$$

term John:

$$w_{tJohn} = Q(\frac{0.10}{0.16}) - Q(\frac{0.07}{0.16}) = 0.1875$$

term Forward:

$$w_{tForward} = Q(\frac{0.16}{0.16}) - Q(\frac{0.10}{0.16}) = 0.3750$$

For Doc.1, *concept Terry* has its satisfaction as:

$$\begin{aligned} s_{cTerry} &= 0.4375 \times 0.3679 + 0.1875 \times 0 + 0.3750 \times 0 \\ &= 0.1610 \end{aligned} \tag{13}$$

The ordered satisfactions of the terms in *Club Chelsea* are:

Concept Player Chelsea	satisfaction	importance	weights
term: Chelsea	0.8465	0.15	0.6000
term: London	0.6065	0.02	0.0800
term: Stamford	0.6065	0.08	0.3200

The satisfaction of the combination of the terms in *Club Chelsea* is:

$$\begin{aligned} s"_{cChelsea} &= 0.8465 \times 0.6000 + 0.6065 \times 0.0800 + 0.6065 \times 0.3200 \\ &= 0.7505 \end{aligned} \tag{14}$$

The ordered satisfactions of the three concepts are:

RC	satisfaction	importance	weights
concept:Chelsea	0.7505	0.35	0.4167
concept:Drogba	0.2130	0.19	0.3167
concept:Terry	0.1610	0.16	0.2667

Therefore, the satisfaction of the RC, which is the satisfaction of HofC_B(some), is:

$$\begin{aligned} s_{cRC} &= 0.4167 \times 0.7505 + 0.3167 \times 0.2130 + 0.2667 \times 0.1610 \\ &= 0.4231 \end{aligned} \tag{15}$$

8.4 Discussion

The results for all cases are summarized in Table 3. There are 12 pieces of information (sum of the frequencies of terms) to satisfy three aimed concepts (Table 1) in both documents. For Doc.1, 10 out of 12 are directly related to *Club Chelsea*, while for Doc.2, 10 out of 12 are directly related to *Player Drogba* and *Player Terry*. In all of the results, the scores for Doc.2 are greater than the scores for Doc.1. This is mainly because different pieces of information have different importance (section 8.2). Moreover, the scores differ when calculated with different hierarchical structures or with different linguistic quantifiers.

Table 3: The results of cases

Case	*Linguistic Quantifier*	*Doc.1*	*Doc.2*
HofC_A(most)	most	0.1914	0.3234
HofC_A(some)	some	0.4231	0.4391
HofC_B(most)	most	0.1977	0.3115
HofC_B(some)	some	0.4231	0.4391

For HoC_A(some) and HofC_B(some) the satisfaction of HofC_A is equal to the satisfaction of HofC_B when the same document is evaluated. That is, with the linguistic quantifier *some*, the hierarchical structures do not affect the calculation of satisfactions. This is because *some* is defined by $Q(r) = r$ (Fig. 6a), which means the aggregation weight of a criterion is decided by its importance value directly. That is, with the linguistic quantifier *some*, the order of aggregation does not change the calculation.

For information aggregated by the linguistic quantifier *most* (defined by $Q(r) = r^2$,Fig. 6b) order is crucial. For example, if there are two pieces of information (i_1 and i_2) with importance $imp_1 = 0.2$ and $imp_2 = 0.3$, respectively, and the order of the aggregation is i_1 then i_2, with *most* their weights are: $wgt_1 = (0.2/0.5)^2 = 0.16$ and $wgt_2 = 1-(0.2/0.5)^2 = 0.84$, respectively. That is, i_2 obtains larger aggregation weight than i_1. If the order of the aggregation is i_2 then i_1, their weights are: $wgt_2^{prime} = (0.3/0.5)^2 = 0.36$ and $wgt_1^{prime} = 1-(0.3/0.5)^2 = 0.64$, respectively. That is, i_1 obtains larger aggregation weight than i_2. Thus with *most*, the order of aggregation is important for calculations. Moreover, because the aggregation of pieces of information is in an order from most satisfied to less satisfied, the less satisfied piece of information tends to have larger weights.

The difference in satisfactions between Doc.1 and Doc.2 with the linguistic quantifier *some* is much smaller than that with the linguistic quantifier *most*. For HofC_A, when the linguistic quantifier changes from *some* to *most*, the difference in satisfactions between Doc.1 and Doc.2 changes from $0.4391 - 0.4231 = 0.0160$ to $0.3234 - 0.1914 = 0.1315$. For HofC_B, when the linguistic quantifier changes from *some* to *most*, the difference in satisfaction between Doc.1 and Doc.2 changes from $0.4391 - 0.4231 = 0.0160$ to $0.3115 - 0.1977 = 0.1138$. This is because satisfaction calculated with *most* is affected by hierarchies while satisfaction with *some* is not.

The case studies are good examples of how HOTIR, especially component Eval, ranks documents. Moreover, the detailed calculations illustrate the effects of the linguistic quantifiers *some* and *most* in HOTIR. In the next chapter, the designed experiments examine the performance of HOTIR with *some* and *most*.

9 Conclusions

The constant need for finding relevant documents leads to increased interests in concept-based search techniques. A necessary part of it is concept identification. In the paper we propose

a human-inspired approach to detect concepts in text documents. The proposed approach is a fusion of multiple techniques: ontology, term importance evaluation schema, hierarchies of concepts, linguistic quantifiers and the aggregation operator OWA. The following elements are proposed for the implementation of the approach:

- *ObKB* – the ontology-based knowledge base provides supplementary knowledge to define concepts in sets of keywords representing users' interests;
- AATI – the scheme for assigning importance values to terms and concepts;
- Hierarchies of Concepts (HofC) – structures built from concepts, terms/attributes; they are used as representation of users' keywords; they are enhanced with knowledge from *ObKB*;
- linguistic quantifiers and OWA – used to aggregate activation levels of concepts and terms from HofC and estimating overall activation of HofC which is equivalent to identification of concepts in documents.

The case study has been designed to show how the proposed approach performs with different settings. The obtained results and inspection of the calculation processes confirm the ability of the approach to identify concepts in documents in a way similar to humans. A new set of experiments is being planed to verify this in real-world environment.

Bibliography

[1] H.M.Haav and T.-L. Lubi, A survery of concept-based information retrieval tools on the web, *Proceedings of 5th East-European conference ADBIS*, Vilnius, Lithuania, 29-41, 2001.

[2] J. A. Gulla and P. G. Auran and K. M. Risvik, Linguistics in Large-Scale Web Search, *in: Natural Language Processing and Information Systems*, 218-222, 2002.

[3] A. Spink and D. Wolfram and M. B. J. Jansen and T. Saracevic, Searching the Web: the public and their queries, *Journal of the American Society for Information Science and Technology*, 52: 226-234, 2001.

[4] V. Vidulin, M. Lustrek and M. Gams, Training a genre classifier for automatic classification of Web pages, *Journal of Computing and Information Technology*, 15(4): 305-311, 2007.

[5] Y. Aphinyanaphongs, I Tsamardinos, A. Statnikov, H. Douglas and C. F. Aliferis, Text categorization models for high-quality article retrieval in internal medicine, *Journal of the American Medical Information Association*, 12(2): 207-216, 2005.

[6] A. Anagnostopoulos, A. Broder and K. Punera, Effective and efficient classification on a search-engine modeling, *Knowledge and Information Systems*, 16(2): 129-154, 2008.

[7] B. Choi and X. Peng, Dynamic and hierarchical classification of Web pages, *Online Information Review*, 28(2): 139-147, 2004.

[8] P. J. Anick, Adapting a full-text information retrieval system to the computer troubleshooting domain, *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 349-358, 1994.

[9] W. A. Woods, 1997, Conceptual indexing: a better way to organize knowledge, *Technical Report: TR-97-61, Sun Microsystems, Inc. Mountain View, CA,USA*. http://research.sun.com/techrep/1997/abstract-61.html,

[10] T. E. Gruber, A translation approach to portable ontology specifications, *Knowledge Acquisition*, 5: 199-220, 1993.

[11] G. A. Miller, WordNet: a lexical database for English, *Communications of the ACM*, 38(11): 39-41, 1995.

[12] E. M. Voorhees, Query expansion using lexical-semantic relations, *Proceedings of the 17th Annual ACM SIGIR conference on research and development in information retrieval*, New York, NY, USA, 61-69, 1994.

[13] Z. Gong, C.-W. Cheang and L. H. U, Web query expansion by WordNet, *Lecture notes in computer science*, 3588, 166-175, 2005.

[14] M. Baziz, M. Boughanem, N. Aussenac-Gilles, and C. Chrisment, Semantic Cores for Representing Documents in IR, *Proceedings of 2005 ACM Symposium on Applied Computing*, Santa Fe, New Mexico, 1011-1017, 2005.

[15] S.G. Kolte and S.G. Bhirud, Word Sense Disambiguation Using WordNet Domains, *First International Conference on Emerging Trends in Engineering and Technology, 2008. ICETET '08*, Nagpur, Maharashtra, 1187-1191, July, 2008.

[16] Y.-B. Kim and Y.-S. Kim, Latent Semantic Kernels for WordNet: Transforming a Tree-Like Structure into a Matrix, *ALPIT '08 International Conference on Advanced Language Processing and Web Information Technology*, 76-80, July, 2008.

[17] K. Knight and R. Whitney, Ontology Creation and Use: SENSUS, Information sciences institute, University of Southern California, http://www.isi.edu/natural-language/resources/sensus.html, 1997.

[18] N. Guarino, C. Masolo and G. Vetere, OntoSeek: Content-based Access to the Web, *IEEE Intelligent Systems*, 14(3): 70-80, 1999.

[19] S. E Lewis, Gene Ontology: looking backwards and forwards, *Genome Biology*, 6(1): 103, 2005.

[20] I. Spasic, E. Simeonidis, H. L. Messiha, N. W. Paton, and D. B. Kell, KiPar, a tool for systematic information retrieval regarding parameters for kinetic modelling of yeast metabolic pathways, *Bioinformatics*, 25(11): 1404-1411, 2009.

[21] D.W. Embley, Towards Semantic Understanding - An Approach Based on Information Extraction Ontologies, *Proceedings of the Fifteenth Australasian Database Conference (ADC2004)*, 3-12, 2004.

[22] H.-M.Muller, E. E. Kenny, and P. W. Sternberg, Textpresso: An Ontology-Based Information Retrieval and Extraction System for Biological Literature, *PLoS Biology*, 2(11): 1984-1998, 2004.

[23] H. B. Styltsvig, Ontology-based information retrieval (PHD Thesis), Denmark, 2006.

[24] Ph. Cimiano, P. Haase, M. Herold, M. Mantel, and P.Buitelaar, LexOnto: A Model for Ontology Lexicons for Ontology-based NLP, *Proceedings of the OntoLex (From Text to Knowledge: The Lexicon/Ontology Interface) workshop at ISWC07 (International Semantic Web Conference)*, Busan, South-Korea, 2007.

[25] M. Morneau, G. W. Mineau, and D. Corbett, LexOnto: A Model for Ontology Lexicons for Ontology-based NLP, *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, Hongkong, China, 449-455, 2006.

[26] D. Vallet, M. Fernandez and P. Castells, An ontology-based information retrieval model, *Proceedings of 2nd European Semantic Web Conference, ESWC 2005*, Grete, Greece, 455-470, June, 2005

[27] O. Dridi and M.B. Ahmed, Building an ontology-based framework for semantic information retrieval: application to breast cancer, 3rd International Conference on Information and Communication Technologies: From Theory to Applications, ICTTA 2008. , April, 1-6, 2008.

[28] H. Cunningham, D. Maynard, K. Bontcheva , V. Tablan, C. Ursu, M. Dimitrov, M.Dowman, N. Aswani, I.Roberts, Y. Li, A. Shafirin and A. Funk, Developing Language Processing Components with GATE, The University of Sheffield, April, 2009, http://gate.ac.uk/sale/tao/index.html.

[29] P. Castells, M. Fernandez and D. Vallet, An Adaptation of the Vector-Space Model for Ontology-Based Information Retrieval, *IEEE Transactions on Knowledge and Data Engineering*, 19(2): 261-272, 2007.

[30] S. L. Tomassen, Searching with Document Space Adapted Ontologies, in *Emerging Technologies and Information Systems for the Knowledge Society*, 5288, 513-522, 2008.

[31] V. Snasel, P. Moravec and J. Pokomy, WordNet Ontology Based Model for Web Retrieval, *Proceedings of the International Workshop on Challenges in Web Information Retrieval and Integration*, Tokyo,Japan, 220-225, April, 8-9, 2005.

[32] G. Antoniou and F. van Harmelen, A Semantic Web Primer(2nd Edition), *The MIT Press, Cambridge, Massachusetts, London, England*, 2008.

[33] B. McBride, RDF Primer, Aug., 2008, http://www.w3.org/TR/REC-rdf-syntax.

[34] R.R. Yager, On ordered weighted averaging aggregation operators in multi-criteria decision making, *IEEE Transactions on Systems, Man and Cybernetics*, 18: 183-190, 1988.

[35] L.A. Zadeh, A computational approach to fuzzy quantifiers in natural language, *Computers and Mathematics with Applications* 9: 149-184, 1983.

[36] R.R. Yager, Families of OWA operators, *Fuzzy Sets and Systems*, 59: 125-148, 1993.

[37] R.R.Yager, A Hierarchical Document Retrieval Language, *Information Retrieval*, 3: 357-377, 2000.

[38] Wikipedia, Power Iteration, May, 2008, http://en.wikipedia.org/wiki/Power_method.

[39] Z.Li and M.Reformat, A Schema for Ontology-based Concept Definition and Identification, *International Journal of Computer Applications in Technology*, 38(4): 333-345, 2010.

[40] D. Ferrucci and A. Lally, UIMA: an architectural approach to unstructured information processing in the corporate research environment, *Natural Language Engineering*, 10(3-4): 327-348, 2004.